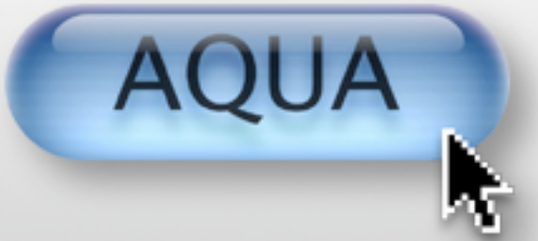
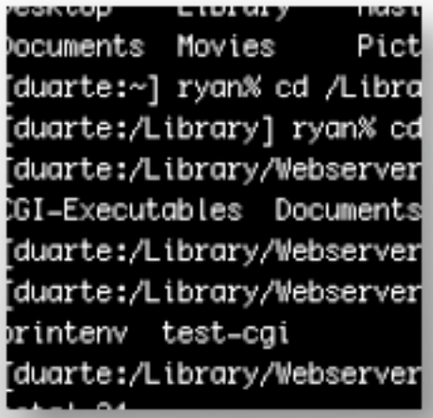
Cocoa 2D Drawing Techniques

Session 413





Cocoa 2D Drawing Techniques

John C. Randolph
Sr. Software Engineer, Apple WWDR

Troy Stephens
Software Engineer, Cocoa Frameworks

What We'll Cover

- The Cocoa classes involved in 2D rendering
- How to draw using the Cocoa frameworks
- Images, paths, and text
- Manipulating the coordinate space
- How the Quartz drawing model is reflected in the Cocoa framework

What We'll Cover

- More sophisticated drawing techniques
  - Combining vector and image drawing
- Changes to the Cocoa 2D drawing API in Panther
  - New APIs
  - Some changes to existing classes
- One or two demos

Where Does This Fit In?

Applications

Aqua

BSD | Java | Carbon | Cocoa

Quartz | OpenGL | QuickTime

Darwin

Where Does This Fit In?

Applications

Aqua

BSD | Java | Carbon | Cocoa

Quartz | OpenGL | QuickTime

Darwin

Where Does This Fit In?

Applications

Aqua

BSD | Java | Carbon | Cocoa

Quartz | OpenGL | QuickTime

Darwin

Quartz 2D Features

- Window management
- Vector drawing
- Images
- Compositing
- Text

Quartz 2D Features: Window Server

- Quartz Compositor
- Hardware-accelerated window compositing
- Transparency
- Windows live in OpenGL texture memory

Quartz 2D Features: Vector Drawing

- Postscript/PDF/SVG-Style drawing model
- Paths that can be filled or stroked
- Resolution-independent
- Anti-aliasing for free

The Quartz 2D Coordinate System

- Coordinates are floating-point values
- On today's displays, whole-number coordinates land between screen pixels
- Don't count on this!
- Check to see whether your drawing tweaks are needed

Quartz 2D Features: Images

- Scaling
- Rotation
- Color-depth resampling
- Color-managed rendering
- DCT decoding (JPEG)
- All Velocity Engine optimized

Quartz 2D Features: Compositing

- Rich set of compositing operators
- 14 different modes
- Compositing uses both *source* and *destination* alpha
- Optimized for Velocity Engine
- /Developer/Examples/AppKit/CompositeLab
- http://developer.apple.com/samplecode/Cocoa/
  - Tinted image
  - Cropped image

Quartz 2D Features: Text

- Glyph generation and rendering
- Also Velocity Engine optimized

AppKit Classes

- Window management
  - NSWindow
- Vector drawing
  - NSBezierPath
- Images and compositing
  - NSImage, NSImageRep
- Text
  - NSText, NSString, NSAttributedString

Where to Draw?

- NSWindow manages a CGWindow
  - Gets events
  - Talks to the window server
  - Has a drawing context
- Most drawing happens in NSView instances
  - Responds to -lockFocus/-unlockFocus
- NSImage is also a place to draw

Review: NSView

- Think of it as “owning” portion of a window
- An NSView has its own coordinate space
  - Not necessarily its own drawing context
- NSViews are hierarchical
- Most drawing happens in the `-drawRect:` method
- Some changes coming for Panther

Review: NSView's Frame and Bounds

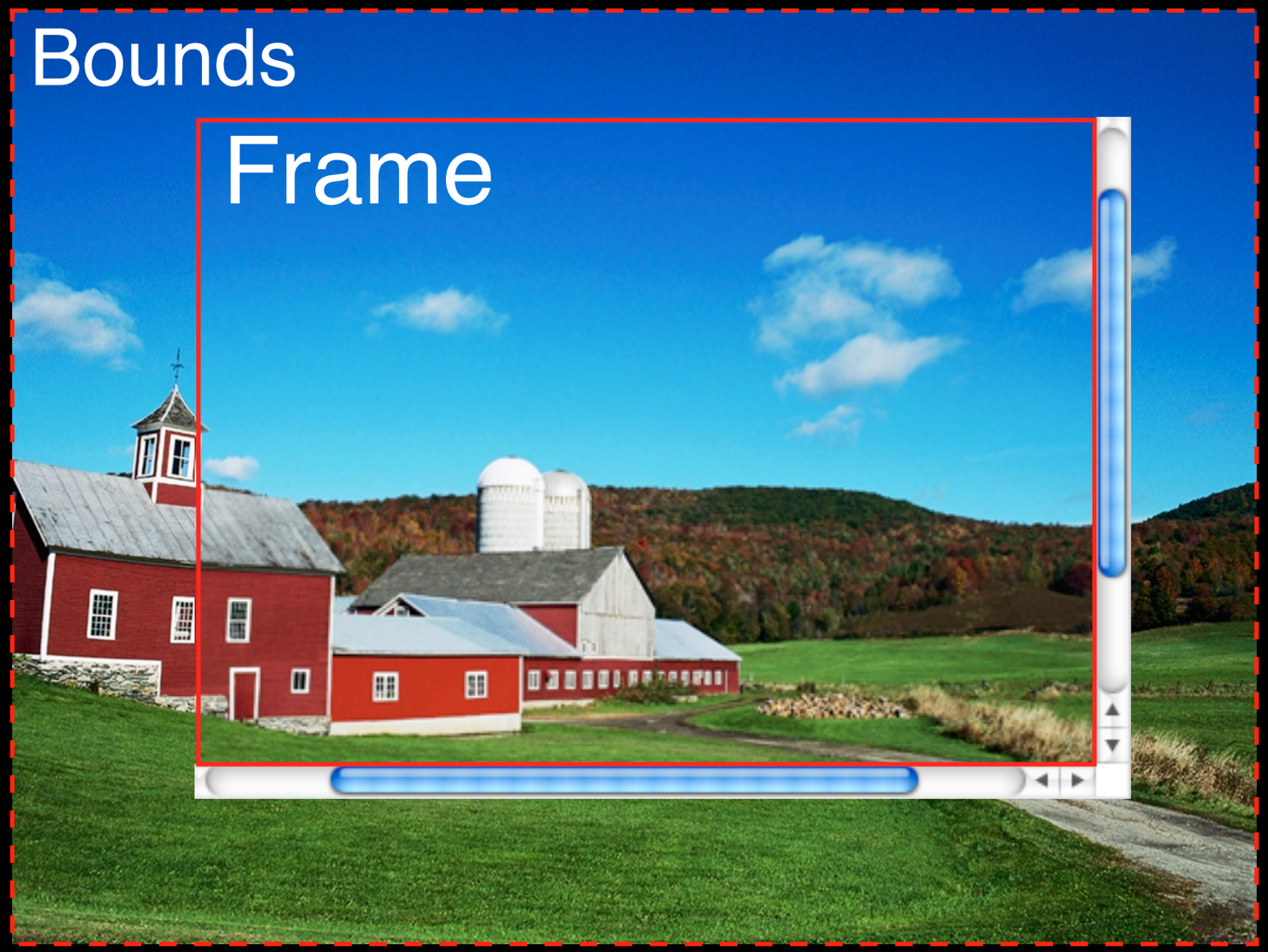
- A View's *frame* determines where it's positioned in its superview
- A View's *bounds* determines its coordinate space
  - Defines the location of the origin
  - Determines the scaling/rotation
  - Manipulate it with `-setBoundsSize:`, `-setBoundsOrigin:`, `-translateOriginToPoint:`, `-rotateByAngle:`, etc.

Review: NSView's Frame and Bounds

Review: NSView's Frame and Bounds

Review: NSView's Frame and Bounds

Review: NSView's Frame and Bounds



NSView Advice

- Avoid calling `-lockfocus`/`-unlockFocus`
  use them when you *must* draw *right away*
- `-setNeedsDisplayInRect:` or `-setNeedsDisplay:`
  are fine in most cases
- Let the framework do the driving!

NSBezierPath for Vector Drawing

- Familiar Path Construction Operations:

  `-moveToPoint:`, `-lineToPoint:`, `-relativeLineToPoint:`, etc.

- Convenience methods

  `+bezierPathWithOvalInRect:`,

- Clipping: `-addClip`, `-setClip`

- Character glyph outlines

- Re-use paths by calling `-removeAllElements`

NSBezierPath (Cont.)

- Bezier cubic splines (of course)
- Line caps, flatness, and so on
- Everything you need from Postscript or PDF

Review: AppKit Drawing Functions

- `NSRectFill()`, `NSRectFillList()`, `NSCopyBits()`, `NSFrameRectWithWidthUsingOperation()`, and many, many more
- Take the time to read about them

Review: NSImage

- High-level abstraction of Images
- Container for NSImageRep instances
- NSImage instances *don't have pixels!*
- *Some* NSImageReps have pixels
- Destination for off-screen drawing

NSImage as a Drawing Destination

- When drawing in an image…
  - The NSView methods for changing the coordinate space are not available
  - …but CG functions still are
  - NSAffineTransform comes in handy here

More NSImage Features

- Progressive display
  - Driven by notifications
- Multi-frame (or page) images
  - Use `-setCurrentPage:` for NSPDFImageRep, and `-setProperty:withValue:` for NSBitmapImageRep

`NSImageCurrentFrameDuration`, `NSImageCurrentFrame`

Compositing

- NSImage compositing methods
  - `-compositeToPoint:fromRect:operation:fraction:`
  - `-drawInRect:fromRect:operation:fraction:`
- The `…ToPoint:` and `…InRect:` methods behave differently!
- To composite a color, use `NSFillRectUsingOperation()`

Demo

Cropped Image

What I Did Wrong In That Demo

- Too much drawing
- I should have re-used the NSBezierPath objects, instead of discarding them
- Didn’t bother with scaling the path

Text

- A pretty big subject
- The AppKit adds drawing methods to NSString and NSAttributedString
- NSCell objects are handy for text.
- For more information on Text, please go to the Cocoa Text session!

More AppKit Classes

- NSAffineTransform
  - Any combination of rotation, scaling, shearing
- NSGraphicsContext
  - Info about your current drawing environment
- NSScreen
  - Info about your displays

NSView

Troy Stephens
Application Frameworks Engineer

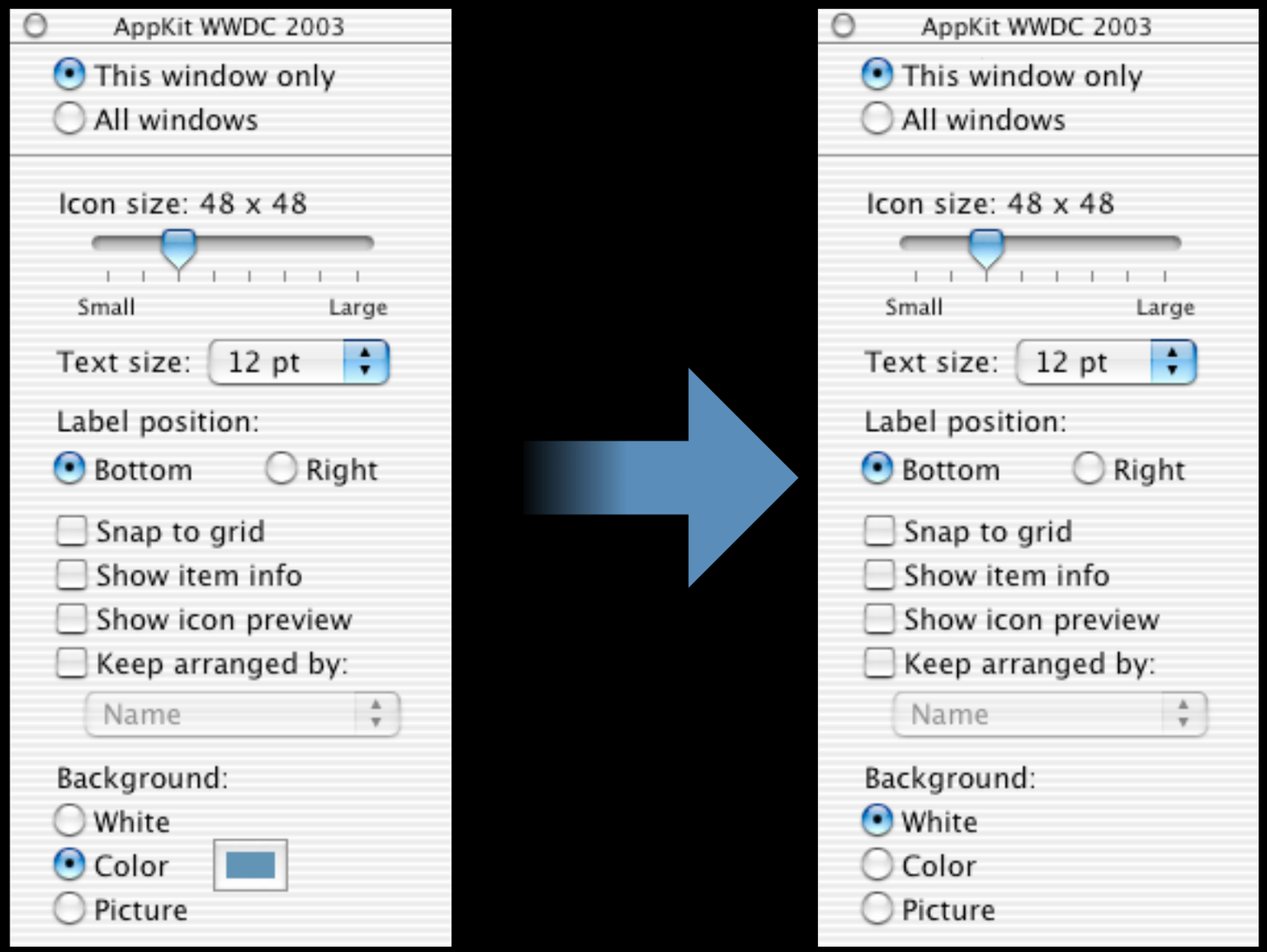
Topics

- Hiding Views
- Optimizing View Drawing

Hiding Views

What if you need to temporarily hide a view?

Hiding Views—an Example



Hiding Views—Jaguar and Earlier

- Remove the view from its superview

```
hiddenView = [aView retain];
[hiddenView removeFromSuperview];
```

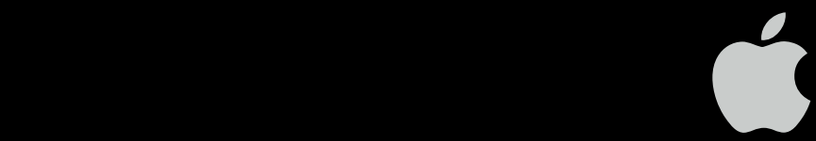
- Re-insert the view to show it again

```
[containerView addSubview:hiddenView];
[hiddenView release];
```

Hiding Views—Jaguar and Earlier

- Some problems with this technique:
  - No autosizing
  - View loses its position in key loop
  - Tooltips, cursor rects, and tracking rects also need managing

```
[view setHidden:YES]
```



Hiding Views In Panther

```
[view setHidden:YES]
```

- View and all descendants disappear
- Stays in parent's subviews list, gets autosized
- Stays in key view loop
- AppKit automatically manages its:
  - Tooltips
  - Cursor rects
  - Tracking rects

Hiding Views In Panther

- Hiding/unhiding

```
- (void) setHidden:(BOOL)hide;
```

- Querying

```
- (BOOL) isHidden;
- (BOOL) isHiddenOrHasHiddenAncestor;
```

Topics

- Hiding Views
- Optimizing View Drawing

Optimizing View Drawing

- What you can do on all system versions
- What Panther can do to help
  - Automatically inherited optimizations
  - Optional API

Mac OS X—All Versions

- Be lazy
  - Avoid invoking `-display...` methods
  - Instead, mark areas as needing drawing:

```
[view setNeedsDisplayInRect:aRect];
[view setNeedsDisplay:YES];
```

  - Dirty only what really needs to be drawn
    - Is `-setNeedsDisplay:YES` the best you can do?
    - Mark individual rects dirty instead of their union

Mac OS X—All Versions

- Draw only what `-drawRect:` is requesting:

```
- (void) drawRect:(NSRect)aRect
{
    ...
    while (thing = [drawableThingEnumerator
nextObject]) {
        if (NSIntersectsRect(aRect, [thing bounds])) {
            [self drawThing:thing];
        }
    }
}
```

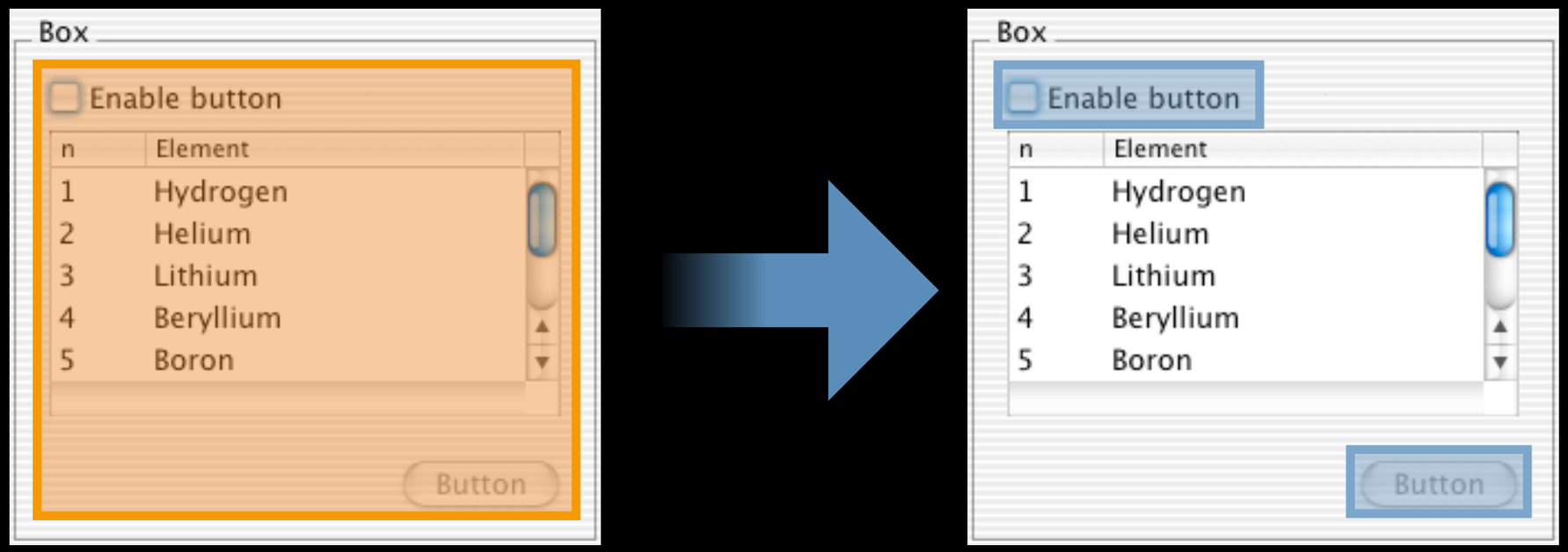
Mac OS X—All Versions

- Override `-isOpaque` if your custom view draws with 100% opaque coverage:

```
- (BOOL) isOpaque
{
    return YES;
}
```

What's New In Panther

- "Needs Display" areas preserved in greater detail
  - Solves the "coalescing problem" in container views
  - Enables views to draw their contents more selectively



What's Not New In Panther

- Rectangles remain the invalidation primitive
- `-drawRect:` remains the basic drawing method for views
- `-drawRect:` is called once each time drawing is needed
- Existing `-drawRect:` implementations require no modification

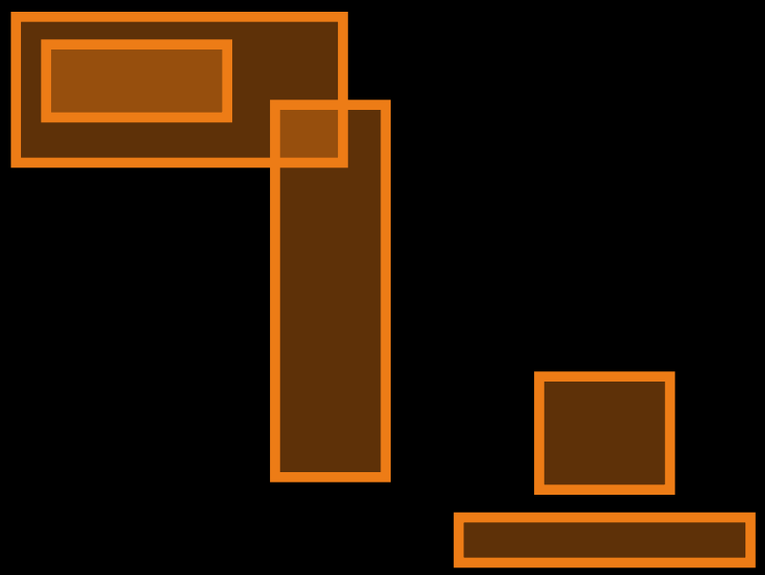
Drawing Less In Panther

```
- (void) getRectsBeingDrawn:(const NSRect **)rects
                      count:(int *)count;
```

- Call this from within -drawRect:
- Returns a list of rectangles that bound the area needing drawing
- Clipping to the area needing drawing is automatic
- -drawRect:'s existing NSRect parameter provides the bounding rectangle of these rects
- Caller doesn't need to worry about "freeing" the rect list

Bounding View Drawing

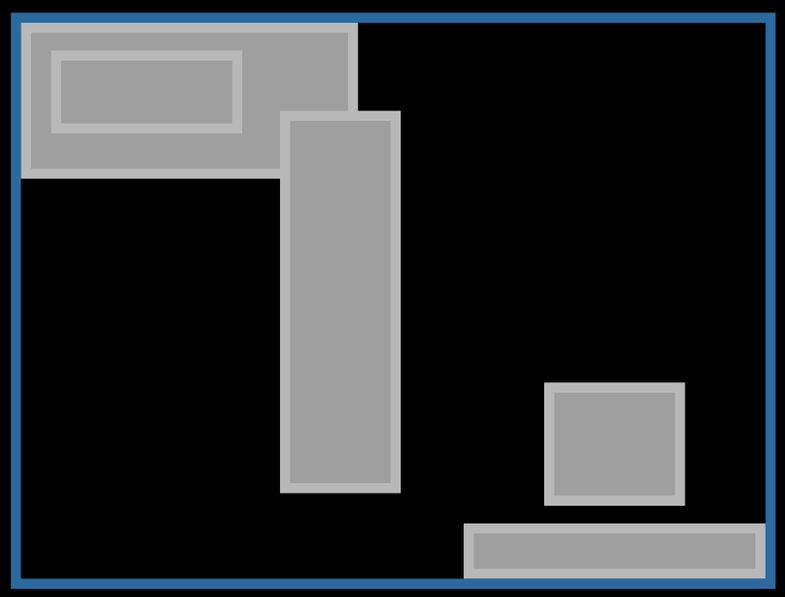
Several rectangles need display.



Jaguar and earlier

Bounding View Drawing

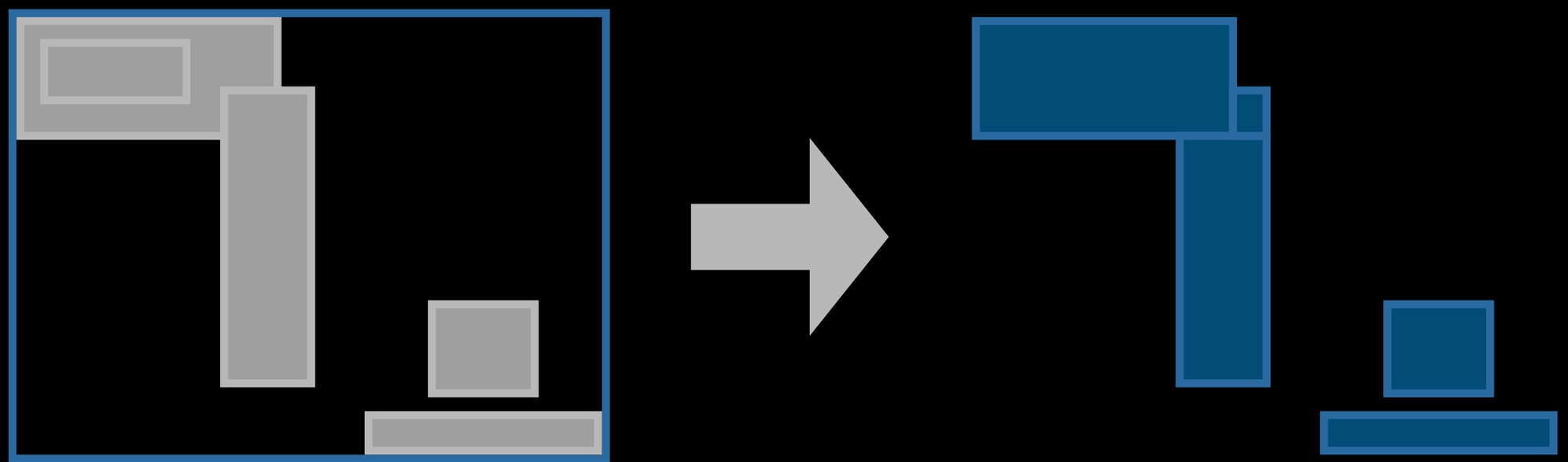
-drawRect: is asked to draw their union.



Jaguar and earlier

Bounding View Drawing

We clip to a list of rects. You need only draw these.

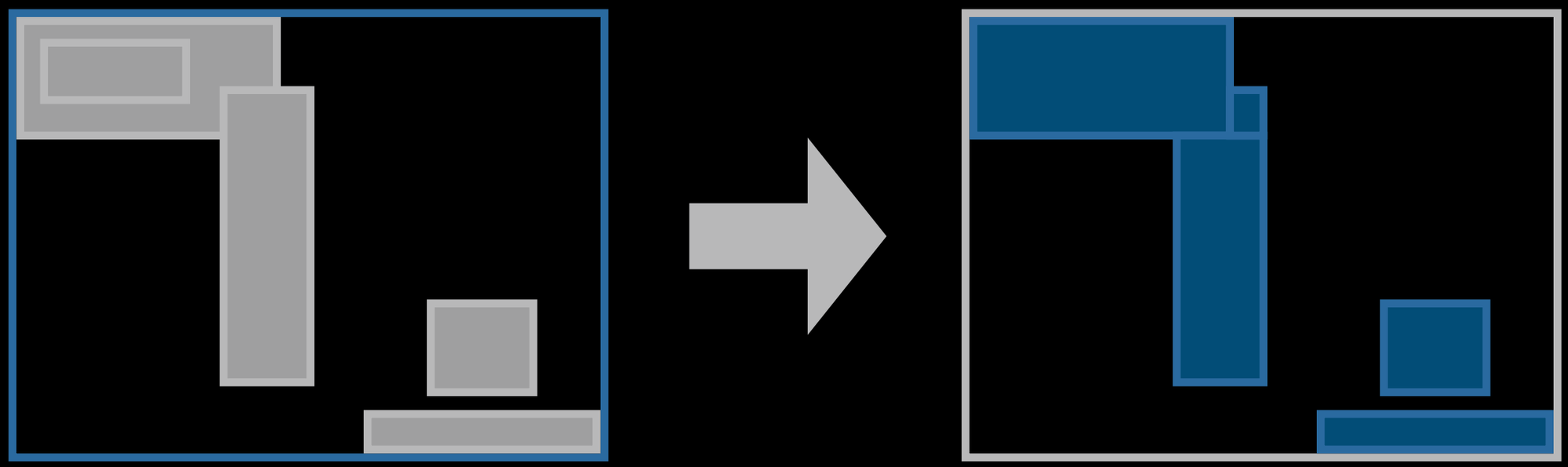


Jaguar and earlier

Panther

Bounding View Drawing

-drawRect: still receives their bounding rect.



Jaguar and earlier

Panther

Example: a View that Draws Objects

```
- (void) drawRect:(NSRect)aRect {
    const NSRect *rects;
    int count, i;
    ...
    [self getRectsBeingDrawn:&rects count:&count];
    while (thing = [thingEnumerator nextObject]) {
        // First do a trivial rejection test.
        if (NSIntersectsRect([thing bounds], aRect)) {
            for (i = 0; i < count; i++) {
                if (NSIntersectsRect([thing bounds], rects[i])) {
                    [self drawThing:thing];
                    break;
                }
            }
        }
    }
}
```

Example: a View that Draws Objects

```
- (void) drawRect:(NSRect)aRect {

    ...

    while (thing = [thingEnumerator nextObject]) {

        if ([self needsToDrawRect:[thing bounds]]) {

                    [self drawThing:thing];

        }
    }
}
```

Example: a View that Draws an Image

- Rect list specifies the image sections we need to composite into the view

Example: a View that Draws an Image

```
- (void)drawRect:(NSRect)rect {
    const NSRect *rectsBeingDrawn;
    int n, rectsBeingDrawnCount;
    NSRect viewSpaceDirtyRect, imageSpaceDirtyRect;

    [self getRectsBeingDrawn:&rectsBeingDrawn count:&rectsBeingDrawnCount];
    for (n = 0; n < rectsBeingDrawnCount; n++) {
        // Intersect rectangle n with the area occupied by the image.
        viewSpaceDirtyRect = NSIntersectionRect(rectsBeingDrawn[n], imageRect);
        // If any part of the image falls within rectangle n...
        if (!NSIsEmptyRect(viewSpaceDirtyRect)) {
            // Determine the corresponding subrect of the image.
            imageSpaceDirtyRect = NSOffsetRect(viewSpaceDirtyRect, -
imageRect.origin.x, -imageRect.origin.y);
            [image compositeToPoint:viewSpaceDirtyRect.origin
fromRect:imageSpaceDirtyRect operation:NSCompositeCopy];
        }
    }
}
```

Example: a View that Draws an Image

- Some observations:
  - Don’t really need the default, AppKit-provided clipping
  - Could avoid the associated setup and enforcement cost

Drawing Without a Net

- Default implementation returns `YES`
- Override to return `NO` if clipping is not required
- View can then provide its own clipping,

…or …draw very carefully!

- One possible implementation strategy:
  - Outer loop over rect list
  - Clip to one rect at a time

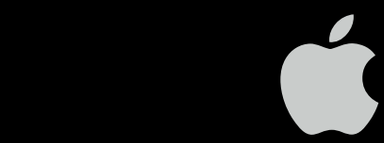
New NSView API Panther

- Hiding Views

```
- (void) setHidden:(BOOL)hide;
- (BOOL) isHidden;
- (BOOL) isHiddenOrHasHiddenAncestor;
```

- Drawing Less

```
- (BOOL) needsToDrawRect:(NSRect)aRect;
- (void) getRectsBeingDrawn:(const NSRect **)rects
                      count:(int *)count;
- (BOOL) wantsDefaultClipping;
```



Conclusion

- We saw how to draw using the classes in the Cocoa framework
- And how to do some sophisticated things with not much code at all
- Cocoa:
  - Simple things simple, complex things possible

Wrap Up

Roadmap

400 Introduction to Developing Applications with Cocoa	Nob Hill Mon., 5:00pm
410 Cocoa Update	Presidio Wed., 10:30am
412 What's New in Cocoa User Interface Programming	Presidio Wed., 3:30pm

Roadmap

419 Cocoa Performance Techniques	Russian Hill Fri., 9:00 am
424 Cocoa: Tips and Tricks	Mission Fri., 2:00 pm
427 What's New in Cocoa Text	Presidio Fri., 5:00 pm
FF008 Cocoa Feedback Forum	North Beach Thu., 10:30 am

Who to Contact

John Randolph
Senior Software Engineer, WWDR SWAT
jcr@.apple.com

Travis Brown
Graphics and Imaging Evangelist
travisb@apple.com

John Geleynse
Manager, Software Evangelism
geleynse@apple.com

Developer Technical Support
Code-level questions (fee-based)
dts@apple.com

http://developer.apple.com/wwdc2003/urls.html

Who to Contact

Cocoa Feedback
For suggestions and feedback
cocoa-feedback@group.apple.com

http://developer.apple.com/wwdc2003/urls.html

For More Information

Reference Library

Documentation

Basic Drawing
ADC Home > Documentation > Cocoa > Graphics & Imaging >

Drawing and Views
ADC Home > Documentation > Cocoa > Graphics & Imaging >

The Drawing Environment
ADC Home > Documentation > Cocoa > Graphics & Imaging >

Using Color
ADC Home > Documentation > Cocoa > Graphics & Imaging >

Reference Library

Documentation (Cont.)

Application Kit Reference for Objective-C
ADC Home > Documentation > Cocoa > Reference >

Application Kit Reference for Java
ADC Home > Documentation > Cocoa > Reference >

Drawing With Quartz 2D
ADC Home > Documentation > Cocoa > Reference >

Reference Library

Sample Code

Sketch, BezierPathLab, CompositeLab, DotView, etc.
/Developer/Examples/AppKit/

Cropped Image, Tinted Image, Polygons, RGB Image, etc.
http://developer.apple.com/samplecode/Sample_Code/Cocoa.htm

Reference Library

Other Resources

Cocoa in a Nutshell (Beam, Davidson)
Building Cocoa Applications (Garfinkel, Mahoney)
Learning Cocoa with Objective-C (Davidson)
Published by O'Reilly & Associates

Cocoa Programming (Anguish, Buck, Yactman)
Published by SAMS

Cocoa Recipes for Mac OS X (Cheeseman)
Published by PeachPit press

Reference Library

Other Resources

Cocoa Programming for Mac OS X (Hillegass)
Published by Addison-Wesley

Cocoa Programming for Dummies
Published by For Dummies

Reference Library

Mailing Lists

Cocoa-Dev, ProjectBuilder-users
Sign up at http://lists.apple.com

MacOSX-Dev
Sign up at http://www.omnigroup.com/developer/mailinglists/

Q&A

Cocoa Frameworks Team

http://developer.apple.com/wwdc2003/urls.html

WWDC 2003

Worldwide Developers Conference

© 2001 and TM Apple Computer, Inc. All rights reserved.